Wasserspeicherkraftwerk

Aufgaben

Stauseen dienen neben der Wasserbereitstellung für die Binnenschifffahrt und dem Schutz flussabwärts liegender Ortschaften vor Hochwasser auch der Gewinnung elektrischer Energie. Bei Wasserspeicherkraftwerken (Material 1) wird die Fallhöhe des von einem Wehr gestauten Wassers zur Energiegewinnung genutzt. Im Kraftwerk wird das Wasser auf Turbinen geleitet, die Generatoren antreiben. Je nach Bedarf und Wasserstand wird Wasser des Sees durch das Wasserkraftwerk geleitet und Strom produziert. Für einen kommunalen Energieversorger sollen eine Datenbank sowie eine Steuerungssoftware für ein Wasserspeicherkraftwerk entwickelt werden.

1 Datenbank für die Prüfung der Messinstrumente
Das Wasserspeicherkraftwerk besitzt fünf Turbinen, drei Schleusen und mehrere Messinstrumente. Die Messinstrumente der Stauanlage müssen regelmäßig geprüft werden. Ein erstes Entity-Relationship-Modell (ERM) ist in Material 2 dargestellt.

1.1 Beschreiben Sie die im ERM auftretenden Notationselemente und erklären Sie deren Bedeutung jeweils anhand eines Beispiels aus dem vorliegenden Modell.

(6 BE)

1.2 Überführen Sie das gegebene ER-Modell in das relationale Modell in der 3. Normalform und begründen Sie Ihre Entscheidungen.

Hinweis: Alle Relationen sind in der Schreibweise `Relation (PK, Attribut, …, FK#)` anzugeben.

(9 BE)

1.3 Einige Daten der Datenbank sollen ausgewertet bzw. geändert werden.

1.3.1 Da alle Temperatursensoren der Firma Ondokei nicht mehr dem geforderten Standard entsprechen, sollen diese deaktiviert werden und langfristig durch Temperatursensoren des Herstellers B+B mit einer garantierten Messanzahl von 2.000.000 Vorgängen ersetzt werden. Für erste Tests soll zunächst nur ein neuer Temperatursensor installiert werden. Zur Überprüfung der Funktionalität wird sofort nach Installation des Messinstruments eine Messung vorgenommen. Formulieren Sie die SQL-Anweisungen, welche die entsprechenden Geräte in der Datenbank auf nicht aktiv setzt, den neuen Temperatursensor (ID 352) hinzufügt und die erste Messung einfügt.

Hinweis: Fehlende Werte sind sinnvoll zu ergänzen.

(5 BE)

1.3.2 Alle Messinstrumente haben eine vom Hersteller garantierte Anzahl an Messvorgängen. Der Betreiber des Stausees möchte sechs Wochen vorher informiert werden, welche aktiven Messinstrumente ausgetauscht werden müssen. Entwickeln Sie die entsprechende SQL-Anweisung.

Hinweis: Jedes Messinstrument ermittelt im Minutentakt Messdaten.

(3 BE)

1.3.3 Für das erste Quartal des Jahres 2023 soll ermittelt werden, welche Messkontrollen noch nicht abgeschlossen wurden. Es sollen für diese noch nicht abgeschlossenen Messkontrollen jeweils die ID und Geräteart des Messinstruments und der vollständige Name der für die Messkontrolle zuständigen Mitarbeiterinnen und Mitarbeiter (im Folgenden Mitarbeiter genannt) ausgegeben werden. Die Liste soll nach Nachname und Vorname der Mitarbeiter sortiert sein. Entwickeln Sie eine entsprechende SQL-Anweisung.

(4 BE)

1.3.4 Die Leistung wird als optimal betrachtet, wenn mindestens 648.000 Kubikmeter Wasser pro Stunde durch die Turbinen laufen. Entwickeln Sie eine SQL-Anweisung, die die Anzahl der Tage im Februar 2023 ermittelt, an denen genügend Wasser im oberen Speicherbecken vorhanden war, um eine optimale Leistung zu gewährleisten.

Hinweise: Das Messinstrument des Pegelstands im oberen Speicherbecken hat die ID 150. Die Funktionen `MONTH` bzw. `DAY` geben den Monat bzw. den Tag des übergebenen Datums zurück.

(4 BE)

1.4 Die Datenbank soll weiteren Anforderungen gerecht werden:

- Es werden Teams gebildet, wobei ein Mitarbeiter als Teamleiter eingesetzt wird.
- Ein Mitarbeiter gehört zu mindestens einem Team. Jeder Mitarbeiter kann Teamleiter mehrerer Teams sein.
- Zusätzlich zu den Messkontrollen werden zwei bis viermal jährlich visuelle Kontrollen durch qualifizierte Mitarbeiter durchgeführt, die im Jahresbericht festzuhalten sind. Eine visuelle Kontrolle wird von einem einzelnen Mitarbeiter durchgeführt. Der Zeitraum der Kontrolle ist zu speichern.
- Bei visuellen Kontrollen werden die fünf Turbinen und drei Schleusen begutachtet.
- Eine Turbine speichert ihren Wirkungsgrad, ihren Aktivitätsstatus, die Anzahl der Startvorgänge und besitzt eine eindeutige Gerätenummer.
- Eine Schleuse besitzt eine Gerätenummer und die Information, ob sie geschlossen ist.

Entwickeln und zeichnen Sie die Erweiterung des ERM auf Grundlage der genannten Angaben.

Hinweis: Es sind nur die Erweiterungen und Veränderungen zum ursprünglichen Modell zu berücksichtigen.

(9 BE)

2 Objektorientierte Entwicklung der Kraftwerksteuerung

In dem Wasserspeicherkraftwerk kommen fünf Turbinen zum Einsatz. Um die Turbinenleistung zu regulieren, können die Leitschaufeln in den Turbinen verstellt werden. Die Turbine kann auch als Pumpe arbeiten. Dies macht man sich in den Pumpspeicher-Kraftwerken zunutze, wo eine Turbine und der Generator häufig zur sogenannten Pumpturbine vereinigt sind, die sich wahlweise auf (stromverbrauchenden) Pumpbetrieb oder (stromerzeugenden) Generatorbetrieb umstellen lässt.

Die Steuerung des Wasserkraftwerks misst die Pegelstände, überwacht und schaltet die Turbinen und regelt die Öffnung der Schleusen. Folgende Funktionalitäten sind zu gewährleisten:

- Bei Niedrigwasser oder bei Hochwasser werden die Turbinen abgeschaltet (Material 3). Bei einem normalen Wasserstand laufen vier Turbinen. Die fünfte dient dem Ersatz, falls eine gestörte Turbine gesperrt und abgeschaltet werden muss.
- Nur bei Hochwasser werden die drei Schleusen in der Staumauer geöffnet.

- Der Treibgutrechen vor der Staumauer soll eine Schädigung der Anlage durch grobe Verunreinigungen im Wasser verhindern. Wenn der gemessene Pegelstand vor dem Rechen höher ist als hinter dem Rechen, ist die Reinigung des Rechens erforderlich.

Die Turbinen, Schleusen und Sensoren zur Messung der Pegelstände sind über eine serielle Schnittstelle mit einem Controller verbunden.
Ein erstes UML-Klassendiagramm finden Sie in Material 4. Die technische Dokumentation des Controllers steht Ihnen in Material 5 zur Verfügung.

2.1 Überführen Sie die Klasse `Turbine` in Anweisungen einer objektorientierten Programmiersprache und implementieren Sie den Konstruktor und die dargestellten Methoden.

Hinweise: Jede Turbine ist mit Betriebsbeginn nicht gesperrt. Im Attribut `laufzeit` wird die Gesamtlaufzeit der Turbine in Minuten erfasst. Hat das Attribut `drehzahlAktuell` den Wert 0, ist die Turbine ausgeschaltet. Im eingeschalteten Zustand wird der Wert der optimalen Drehzahl der aktuellen Drehzahl zugewiesen. Die Dokumentation der Klasse `DateTime` finden Sie in Material 6.

(6 BE)

2.2 Implementieren Sie die Klasse `Controller` mit dem Konstruktor sowie den Methoden `schalte(turbine: Turbine)` und `lesePegelstand(channel: char)`.

Hinweise: Die im Klassendiagramm dargestellten Konstanten müssen nicht implementiert werden. Die technische Dokumentation zum Controller in Material 5 ist zu berücksichtigen. Die Dokumentation der Klasse `Serial` ist in Material 6 zu finden.

(8 BE)

2.3 Die Messung der Wasserstände und die Erfassung von Störungen der Turbinen laufen jeweils in einem eigenen Thread. In dem UML-Sequenzdiagramm in Material 7 ist das Zusammenspiel des `Steuerung`-Objekts mit den Thread-Objekten der Klassen `Rechenanlage` und `Stoerungsmeldung` abgebildet.

2.3.1 Erläutern Sie den Zweck von Sequenzdiagrammen und beschreiben Sie auf der Grundlage von Material 7 die Bedeutung der Komponenten anhand von Beispielen.

(6 BE)

2.3.2 Implementieren Sie die Klasse `Rechenanlage` unter Berücksichtigung des Klassendiagramms (Material 4) und des Sequenzdiagramms (Material 7) ohne die private Methode `starteReinigung()`.

Hinweis: Aus dem Attribut `pegelstandVor` wird gemäß Tabelle in Material 3 die Wasserstandstufe abgeleitet.

(8 BE)

2.4 Im Konstruktor der Klasse `Steuerung` werden die Objekte gemäß Sequenzdiagramm (Material 7) erzeugt und die Threads gestartet. Außerdem wird ein Array für die Turbinen angelegt und die entsprechende Anzahl von Objekten instanziiert, wobei die Nummerierung der Turbinen bei 1 beginnt.
Die Methode `schalten()` erhält als Parameter die ermittelte Wasserstandstufe und regelt die Zustände der Turbinen und Schleusen gemäß den Angaben in Material 3.
Die Ein- bzw. Ausschaltsignale werden über die Methoden der Klasse Controller realisiert.

2.4.1 Implementieren Sie die Klasse `Steuerung` mit ihrem Konstruktor und der Methode `schalten()`.

(10 BE)

2.4.2 Die Methode `behandleStoerung()` der Klasse `Steuerung` sperrt die Turbine mit der entsprechenden Turbinennummer. Wenn sie eingeschaltet ist, wird sie ausgeschaltet. Die Ersatzturbine, wenn möglich, wird eingeschaltet.
Entwickeln und zeichnen Sie ein Struktogramm für diese Methode.

(6 BE)

2.5 Die Fernsteuerung des Wasserkraftwerks soll über eine Bedienoberfläche durch einen Mitarbeiter erfolgen, wobei folgende Anforderungen zu berücksichtigen sind:
- Er kann sich unter Angabe des Namens und Passworts anmelden und sich auch wieder abmelden.
- Er kann die Betriebsart (Pumpbetrieb oder Generatorbetrieb) einstellen.
- Er kann die Turbinendrehzahl und die Winkel der Leitschaufeln einstellen. Dazu muss er die aktuelle Durchflussrate ermitteln.
- Der Mitarbeiter kann die Messung erlauben und Messwerte prüfen.
- Wenn die Messwerte eine Abweichung von mehr als einem Prozent zu den Referenzgeräten aufweisen, kann er die Messerlaubnis entziehen, um Messfehler zu verhindern.
- Er kann die Anlage hoch- und herunterfahren.

2.5.1 Entwickeln und zeichnen Sie ein UML-Anwendungsfalldiagramm für die genannten Funktionalitäten.

(8 BE)

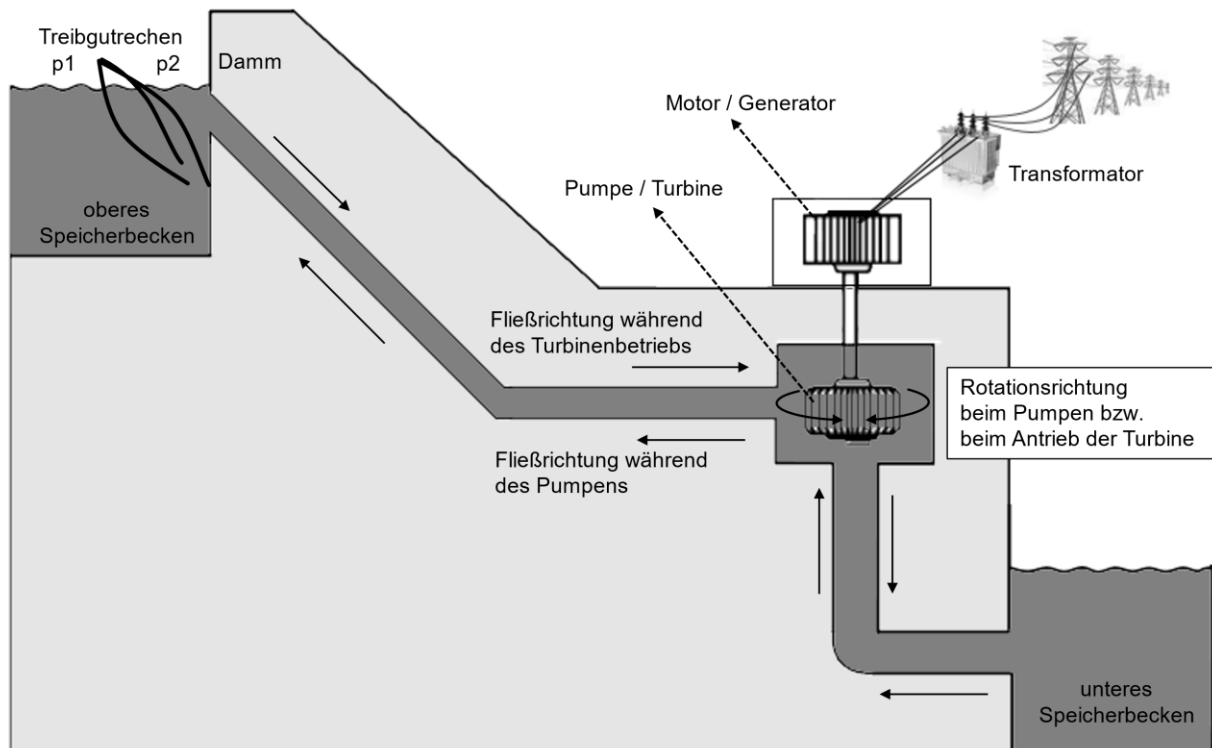
2.5.2 Das Klassendiagramm in Material 4 soll um die Funktionalitäten aus Aufgabe 2.5 zur Fernsteuerung der Kraftwerksanlage erweitert werden.
Modellieren und zeichnen Sie das Klassendiagramm in UML-Notation mit Attributen Konstruktoren, Methoden und Beziehungen.

Hinweise: Die Attribute der Messinstrumente, Messwerte und Mitarbeiter sind dem ERM (Material 2) zu entnehmen. Die Fernsteuerung kann zeitgleich nur von einem Mitarbeiter ausgeführt werden. Get- und set-Methoden müssen nicht dargestellt werden. Eine Vorlage ist in Material 8 zu finden.

(8 BE)

Material 1

Wasserspeicherkraftwerk (schematisch)[1]

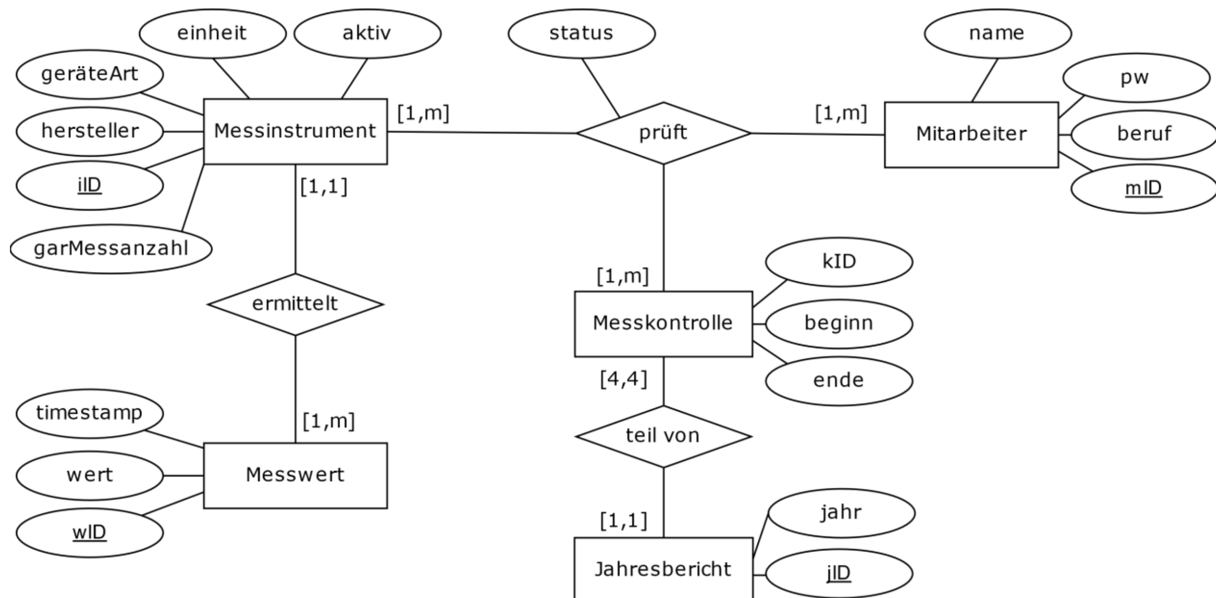


Hinweis: Dargestellt ist hier der Betrieb zur Stromgewinnung. p1 bezeichnet den Pegelsensor vor, p2 den Pegelsensor nach dem Treibgutrechen. Weichen die Pegelstände vor und nach dem Rechen um den Wert 1 voneinander ab, wird die Reinigung des Rechens gestartet.

[1] Eigene Abbildung nach: Nikolaidis, P./ Poullikkas, A. (2017): Schematic diagram of pumped hydro storage plant. In: Journal of Power Technologies 97. S. 220-245. Unter: https://www.researchgate.net/profile/Pavlos-Nikolaidis/publication/320755664/figure/fig3/AS:555773856055296@1509518221155/Schematic-diagram-of-pumped-hydro-storage-plant.png (verändert) (abgerufen am 10.01.2022).

Material 2

Entity-Relationship-Modell



Hinweise: Das Attribut `garMessanzahl` des Entitätstyps `Messinstrument` gibt die vom Hersteller garantierte Mindestanzahl von Messungen an. Das Attribut `status` der Beziehung `prüft` kann die Werte `laufend` oder `abgeschlossen` besitzen. Das Attribut `geräteArt` kann die Werte `temperatur`, `pegel` oder `durchfluss` besitzen. Jede Mitarbeiterin bzw. jeder Mitarbeiter ist für mindestens ein Messinstrument zuständig. Alle angegebenen IDs werden automatisch inkrementiert.

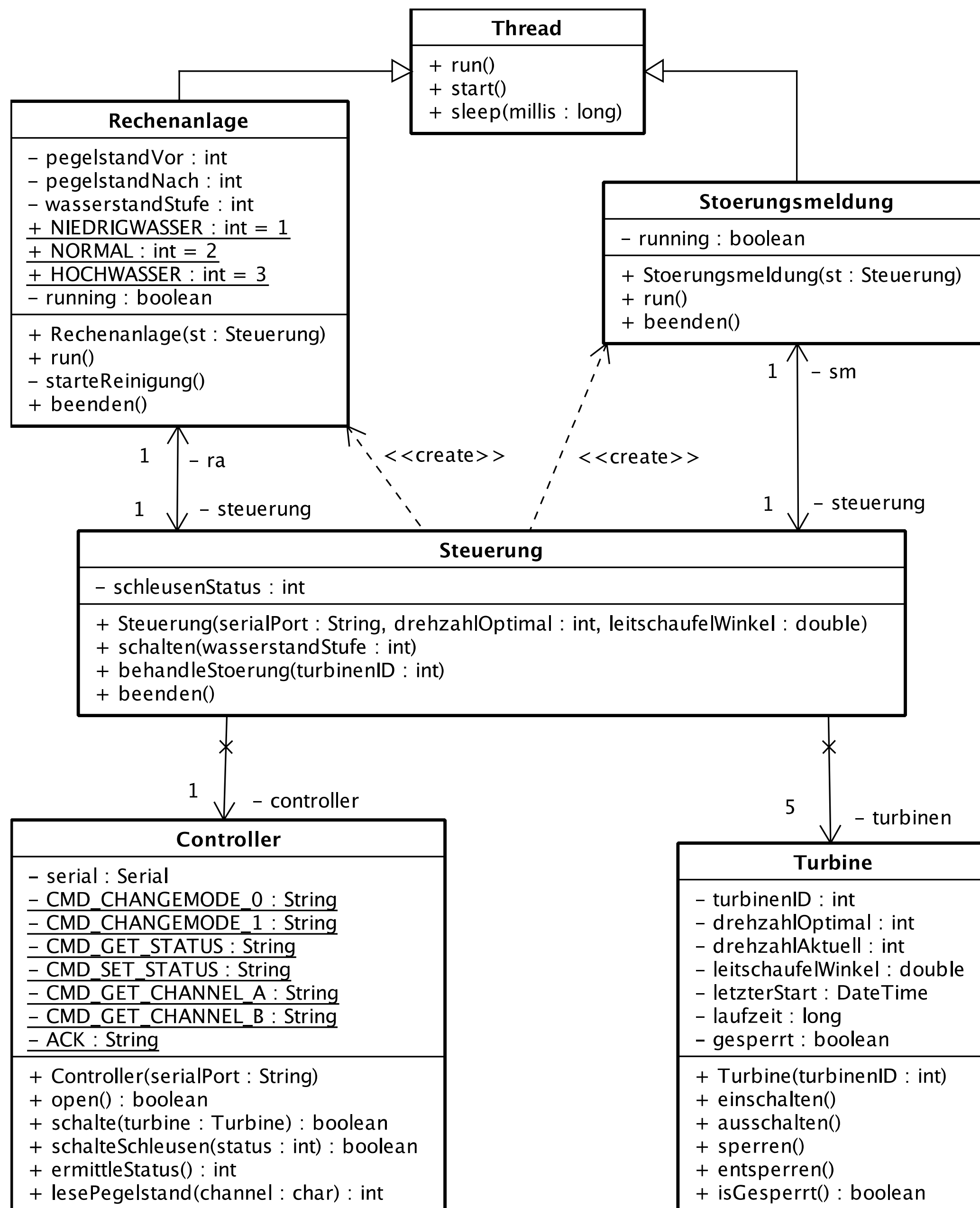
Material 3

Zustand Wasserspeicherkraftwerk nach Wasserständen

Pegelstand in %	Wasserstandsstufe	Zustand der Turbinen	Zustand der Schleusen
kleiner 25	1 (Niedrigwasser)	alle Turbinen sind ausgeschaltet	geschlossen
25 bis 85	2 (Normal)	4 Turbinen laufen	geschlossen
größer 85	3 (Hochwasser)	alle Turbinen sind ausgeschaltet	geöffnet

Material 4

UML-Klassendiagramm



Hinweise:

Klasse `Controller`:

– Die Methode `schalte(turbine: Turbine): boolean` setzt den aktuellen Status der als Parameter übergebenen Turbine. Bei Erfolg gibt sie `true` zurück, ansonsten `false`.
– Die Methode `schalteSchleusen(status: int): boolean` setzt den übergebenen Status der Schleusen. Bei Erfolg gibt sie `true` zurück, ansonsten `false`.
– Die Methode `ermittleStatus(): int` liefert den Status der Turbinen. An den gesetzten Bits kann die Störung jeder einzelnen Turbine erkannt werden.
– Die Methode `lesePegelstand(channel: char): int` liefert den Wert des analogen Eingangs des übergebenen Channels A oder B.

Get- und set-Methoden sind nicht dargestellt.

Material 5

Technische Dokumentation des Controllers

Datenübertragung: 19200 Baud, 8 Datenbits, 2 Stoppbits, keine Parität

Befehlsübersicht

Befehl	Antwort	Erläuterung
CMD_CHANGEMODE_0	ACK	Change Mode Port 0: Ändern des Betriebszustands einer Turbine; muss vor CMD_SET_STATUS aufgerufen werden
CMD_CHANGEMODE_1	ACK	Change Mode Port 1: Ändern des Zustands der Schleusen; muss vor CMD_SET_STATUS aufgerufen werden
CMD_GET_STATUS	<status>	Das nächste eingelesene Byte liefert den Status der Turbinen an den digitalen Eingängen D0 bis D4: Ein 1-Signal auf einer Datenleitung steht für die Störung der entsprechenden Turbine.
CMD_SET_STATUS	ACK	Nach Senden des Befehls CMD_CHANGEMODE_0: Setzen des Status einer Turbine (Beschreibung siehe unten) Nach Senden des Befehls CMD_CHANGEMODE_1: Setzen des Status der Schleusen (Status 0 schließen, 1 öffnen)
CMD_GET_CHANNEL_A	<wert>	liefert den nächsten eingelesenen Wert vom analogen Eingang A
CMD_GET_CHANNEL_B	<wert>	liefert den nächsten eingelesenen Wert vom analogen Eingang B

Beispiel eines Befehlsaufbaus für das Setzen des Status einer Turbine:
```
CMD_SET_STATUS 1;300;35.0<LF>
```

Allgemein:
```
CMD_SET_STATUS <turbinenNr>;<drehzahlAktuell>;<leitschaufelWinkel><LF>
```

Hinweis: <LF> steht für das Zeilenendezeichen und entspricht dem Zeichen `"\n"`.

Material 6

Klassendokumentationen

Klasse `DateTime`

DateTime
+ DateTime() + isEqual(dt : DateTime) : boolean + isBefore(dt : DateTime) : boolean + isAfter(dt : DateTime) : boolean + until(dt : DateTime) : long + toString() : String

`DateTime()`
erzeugt ein `DateTime`-Objekt mit dem aktuellen Systemdatum und der Systemuhrzeit, im Format dd.mm.yyyy hh:mm.

`isEqual(dt: DateTime): boolean`
liefert `true`, wenn das `DateTime`-Objekt gleich dem des Parameters `dt` ist.

`isBefore(date: DateTime): boolean`
liefert `true`, wenn das `DateTime`-Objekt vor dem des Parameters `dt` liegt.

`isAfter(date: DateTime): boolean`
liefert `true`, wenn das `DateTime`-Objekt nach dem des Parameters `dt` liegt.

`until(dt: DateTime): long`
liefert die Anzahl der zwischen dem `DateTime`-Objekt und dem Parameter `dt` liegenden Minuten.

`toString(): String`
liefert eine String-Repräsentanz des `DateTime`-Objekts im Format dd.mm.yyyy;hh:mm.

Klasse `Serial`

Serial
- portName: String - baudrate: int - dataBits: int - stopBits: int - parity: int
+ Serial(String portName, int baudrate, int dataBits, int stopBits, int parity) + open(): boolean + close() + dataAvailable(): int + read(): byte + read(b: byte[], len: int): int + readLine(): String + write(value: int) + write(b: byte[], len int) + write(s: String)

Ein Exemplar der Klasse `Serial` ermöglicht die Kommunikation über die serielle Schnittstelle

`Serial(…)`
der Konstruktor initialisiert die serielle Schnittstelle ohne sie zu öffnen.

`portName:`	Name des Ports
`baudrate:`	Baudrate
`dataBits:`	Datenbitanzahl
`stopBits:`	Stopbitanzahl
`parity:`	Parität

`open(): boolean`
öffnet die serielle Schnittstelle; liefert `true`, wenn die Schnittstelle verwendbar ist.

`close()`
schließt die serielle Schnittstelle; die Schnittstelle ist dann solange nicht mehr verwendbar, bis sie wieder geöffnet wird.

`dataAvailable(): int`
liefert die Anzahl der Bytes, die von der seriellen Schnittstelle gelesen werden können.

`read(): byte`
liest ein Byte (0..255) von der serielle Schnittstelle, bzw. –1, wenn die Schnittstelle nicht geöffnet ist. Die Methode blockiert, bis ein Byte verfügbar ist.

Material 6 (Fortsetzung)

`read(b: byte[], len: int): int`
liest mehrere Bytes von der seriellen Schnittstelle in ein Byte-Array; liefert die Anzahl der gelesenen Bytes, bzw. –1, wenn der Schnittstelle nicht geöffnet ist. Die Methode blockiert, bis mindestens ein Byte verfügbar ist.

`readLine(): String`
liest eine Zeile von der seriellen Schnittstelle, bzw. liefert `null`, wenn die Schnittstelle nicht geöffnet ist. Eine Zeile wird durch ein Zeilenendezeichen abgeschlossen; das Zeilenendezeichen wird jedoch nicht in den zurückgegebenen String übernommen. Die Methode blockiert, bis eine komplette Zeile eingelesen ist.

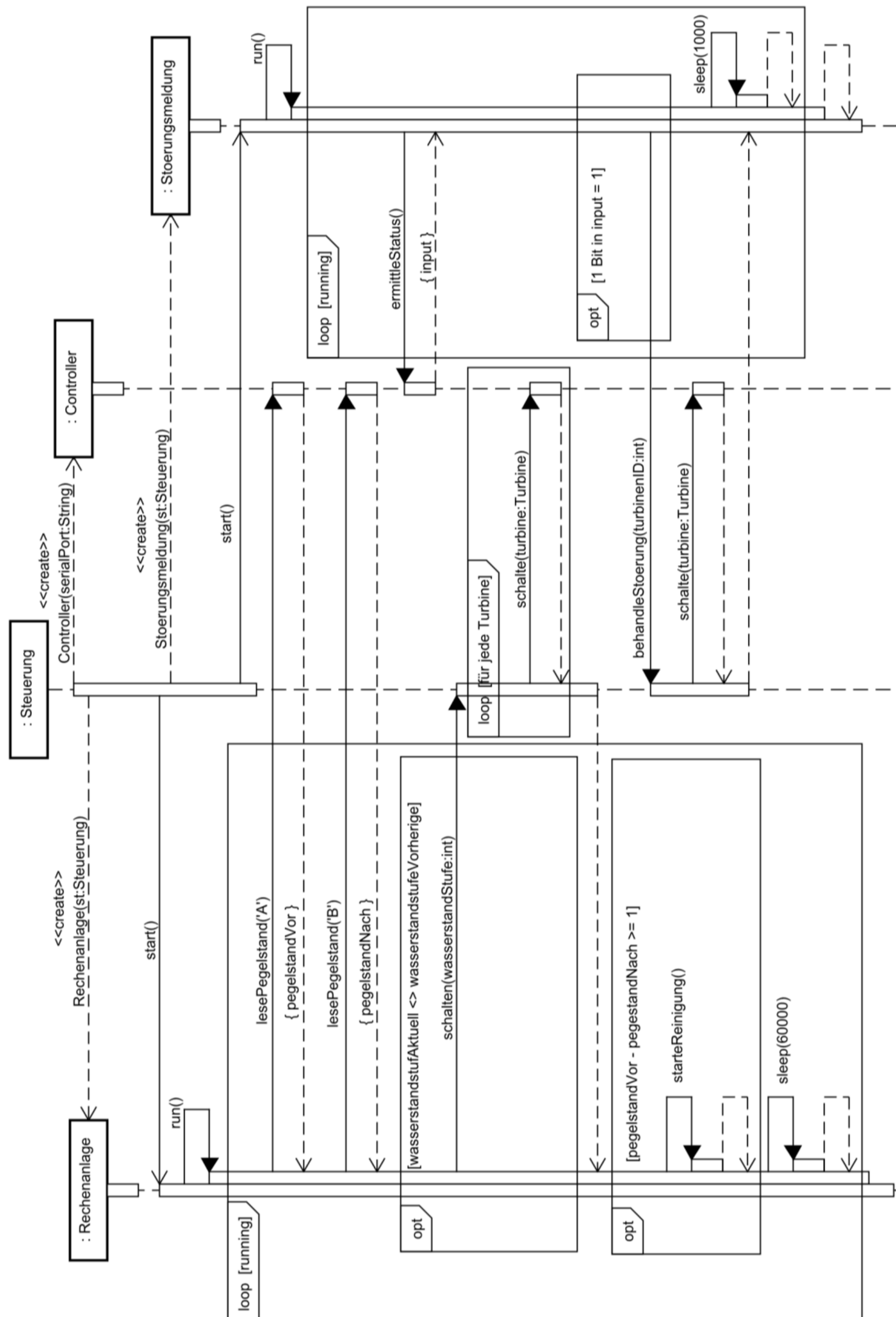
`write(value: int)`
schreibt ein Zeichen auf die serielle Schnittstelle; ist die Schnittstelle nicht geöffnet geschieht nichts.

`write(b: byte[], len: int)`
schreibt mehrere Bytes auf die serielle Schnittstelle; ist die Schnittstelle nicht bereit geschieht nichts.

`write(s: String)`
schreibt einen String auf die serielle Schnittstelle; ist die Schnittstelle nicht geöffnet geschieht nichts.

Material 7

UML-Sequenzdiagramm



Hinweise: Im UML-Sequenzdiagramm ist nur die Kommunikation des `Steuerung`-Objekts mit den Thread-Objekten `Rechenanlage` und `Stoerungsmeldung` sowie dem `Controller`-Objekt dargestellt. Auf die Darstellung der Erzeugung der Turbinen sowie das Schalten der Schleusen wurde hier aus Gründen der Übersichtlichkeit verzichtet.

Material 8

Vorlage Klassendiagramm

